CMSC 430, Practice Problems 1

1. Consider the following grammar: S → S and S | S or S | (S) | true | false
   a. Compute First sets for each production and nonterminal
   b. Explain why the grammar cannot be parsed by a LL(1) parser

2. Consider the following grammar: S → abS | acS | c
   a. Compute First sets for each production and nonterminal
   b. Show why the grammar cannot be parsed by a LL(1) parser.

3. Consider the following grammar: S → Sa | Sc | c
   a. Show why the grammar cannot be parsed by a LL(1) parser.
   b. Rewrite the grammar so it can be parsed by a LL(1) parser.
   c. Using FIRST and FOLLOW , build the LL(1) table for the new grammar
   d. Using the LL(1) table, show how a table-driven parser parses the string "caca"

4. Consider the following grammar

Goal	→	Expr
Expr	→	Term + Expr
	|	Term – Expr
	|	Term
Term	→	Factor * Term
	|	Factor / Term
	|	Factor
Factor	→	num
	|	id

   a. Apply left factoring to create a LL(1) grammar
   b. Compute FIRST and FOLLOW for the modified grammar
   c. Using FIRST and FOLLOW , build the LL(1) table for the new grammar

5. Bottom up parsing
   a. Describe the difference between bottom-up and top-down parsing with respect to a grammar production A → β
   b. Define handle
   c. Explain why bottom-up parsers result in right-most derivations in reverse.
   d. Given the following ACTION/GOTO table, perform a parse of "a", assuming 0 is the start state. Show the contents of the stack, remaining input, and action performed at each step of the shift-reduce parse.

State	Action		Goto	
	a	$	A	B
0	shift 1	reduce $B \rightarrow \epsilon$	2	3
1	shift 3	reduce $A \rightarrow a$	0	
2		accept	3	0
3	shift 1	accept		1

6. LR parsing
   a. What is the canonical set of LR(k) items for a grammar?
   b. What are ACTION/GOTO tables of an LR(k) parser for a grammar?
   c. Given an LR(k) item [$A \rightarrow \alpha \bullet \gamma,\ \delta$] from a production $A \rightarrow \beta$, describe the role of A, α, γ, δ
   d. Given an LR(k) item [$A \rightarrow \alpha \bullet \gamma,\ \delta$], what lookahead allows a shift to be performed (ignoring any other LR(k) items added to the state by closure)?
   e. Given an LR(k) item [$A \rightarrow \alpha \bullet \gamma,\ \delta$], what LR(k) items must be added to the closure?
   f. If the parser is in a state containing an LR(k) item [$A \rightarrow \alpha \bullet \gamma,\ \delta$], what must be at the top of the stack, and why?
   g. Describe what constitutes a shift/reduce conflict
   h. Give an example of two LR(1) items causing a reduce/reduce conflict.
   i. Explain why there are no shift/shift conflicts

7. LALR parsing.
   Consider the following sets of LR(1) items in the states of a LR(1) parser.

   State 0:
   [A → • a , b]
   [A → a • , c]
   [B → a • , b]

   State 1:
   [A → • a , a]
   [A → • a , b]
   [B → a • , b]

   State 2:
   [A → • a , c]
   [A → a • , b]
   [B → a • , a]

   State 3:
   [A → • a , b]
   [B → • a , b]

   a. Find all shift/reduce and reduce/reduce conflicts. List the LR(1) items and lookaheads causing conflicts.
   b. What states would be merged in a LALR(1) parser?
   c. Are any new reduce/reduce conflicts introduced in the LALR(1) parser?
   d. Explain why LALR(1) parsers will not introduce new shift/reduce conflicts.
   e. What is the relationship between LALR(1) and SLR(1) parsers?

8. Consider the following grammar: S → abS | acS | c
    a. Compute FOLLOW(S)
    b. Compute the canonical set of LR(0) items for the grammar
    c. Build the ACTION/GOTO table using LR(0) + FOLLOW
    d. Explain why the grammar is or is not SLR(1)
    e. Use the ACTION/GOTO table to parse the string "abc"

9. Consider the following grammar: S → Sa | Sc | c
    a. Compute the canonical set of LR(1) items for the grammar
    b. Build the ACTION/GOTO table for the grammar
    c. Explain why the grammar is or is not LR(1)
    d. Use the ACTION/GOTO table to parse the string "caca"

10. Consider the following grammar: S → S + S | S * S | (S) | num
    a. Compute the canonical set of LR(1) items for the grammar
    b. Build the ACTION/GOTO table for the grammar
    c. Explain why the grammar is or is not LR(1)
    d. Explain how to use the ACTION/GOTO table so that
        i. * has higher precedence than + and*, + are both left associative
        ii. * has higher precedence than +, * is left associative, and + is right associative
        iii. + has higher precedence than * and *, + are both right associative

11. Consider the following grammar: S → AS | b A → SA | a
    a. Compute the canonical set of LR(0) items for the grammar
    b. Build the ACTION/GOTO table for the grammar
    c. Compute the canonical set of LR(1) items for the grammar
    d. Build the ACTION/GOTO table for the grammar

12. Consider the following grammar: S → S ; a | a
    a. Is the grammar SLR(1)? Is it LR(1)?

13. Consider the following grammar: S → aAd | bBd | aBe | bAe A → c B → c
    a. Is the grammar LR(1)? Is it LALR(1)?

14. Consider the following grammar: S → Aa | bAc | Bc | bBa A → d B → d
    a. Is the grammar LR(1)? Is it LALR(1)?